LA-UR-03-2074
Approved for public release;
distribution is unlimited.

Title: **Approximating the Connectivity between Nodes when Simulating Large-scale Mobile Ad Hoc Radio Networks**

Author(s): Christopher L. Barrett, Madhav V. Marathe, D. Charles Engelhart and Anand Sivasubramaniam

Submitted to:

http://lib-www.lanl.gov/cgi-bin/getfile?00937099.pdf

Approximating the Connectivity between Nodes when Simulating Large-scale Mobile Ad Hoc Radio Networks*

Christopher L. Barrett[1]
Madhav V. Marathe[1]
Los Alamos National Laboratory
Los Alamos NM 87545
Email: barrett,marathe@lanl.gov

D. Charles Engelhart[1,2]
Anand Sivasubramaniam
Penn. State University
University Park PA 16802
Email: engelhar,anand@cse.psu.edu

Abstract

Simulation is a widely used technique in the design and evaluation of mobile ad-hoc networks. However, time and space constraints can often limit the extent of the size, complexity and detail of the networks that can be simulated. Approximations in the system model can possibly alleviate this problem, but we need to be careful about how much accuracy is compromised when employing them.

This paper specifically focuses on one aspect of simulation cost that is incurred in the computation of the connectivity graph that is used to describe what mobile nodes can communicate with whom. Since such a graph is re-computed frequently during the simulation, we explore alternatives to computing this graph exactly and their accuracy in capturing the actual graph properties.

We investigate three approximation alternatives to compute graph connectivity, and propose metrics for expressing their deviations from the actual graph. In addition, the graphs generated by these approximations are compared to the original by examining two previously proposed graph measures - the degree and clustering coefficient distributions. Such comparisons are conducted not only with static graphs, but also with dynamically changing graphs that are a consequence of clients moving. Results indicate that these approximations can be quite effective in avoiding repeated calculation of exact graph connectivity.

*This is a substantially extended and revised version of an earlier paper that appeared in the Proceedings of the 2003 Annual Simulation Symposium. The extensions include the addition of more experimental results to strengthen the arguments within the paper.

[1]The work is supported by the Department of Energy under Contract W-7405-ENG-36.

[2]Work done while the author was a graduate research student in the Basic and Applied Simulation Science Group at Los Alamos National laboratory.

1 Introduction

Simulation plays a crucial role in the design and evaluation of networking protocols, routing mechanisms, and mobility considerations of ad hoc wireless networks. Since these networks are still in their infancy, there are numerous parameters to study. Many of these parameters are virtually impossible to investigate on a large scale in a real deployment and it is also infeasible to model all the complex interactions analytically without loss in accuracy. Simulators serve as convenient non-intrusive tools for studying such networks, and many such tools (e.g. [19, 11]) are in wide use today by several researchers.

A main problem with simulation is the high cost incurred in studying large mobile systems, some of which may even be impossible to model because of their high storage requirements and processing time costs. As the number of nodes in the network grows, routing protocols need to determine the evolving connectivity (who can communicate with whom) between these nodes, which can grow quadratically (we will call this graph connectivity in the rest of this paper). It has been shown [16] that the cost of simulating the ad hoc routing protocol itself does not grow as fast.

For example, if we take the naive approach of computing all $\binom{n}{2}$ connections where each connection computation only takes 100 cycles on 1Ghz machine, computing the entire graph for just one time interval takes 0.049 seconds for a 1000 node system, 499 seconds for a 100,000 node system and is expected to take over 57 days for a ten million node system. This time consuming process not only mandates optimization in the number of connection computations, but also simplifying (the arithmetic) involved in a single connection computation (the constant that is involved with the order notation).

Further, the graph connectivity needs to be recomputed as nodes move, leading to severing existing links and at the same time forming new connections. Thus, it is important to develop better (more scalable) graph connectivity determination techniques in order to simulate much larger networks than is possible with current simulators [19, 11]. In particular, we need to determine the set of neighbors that a node can communicate with at each instant to broadcast messages to them (which many protocols rely on).

Previous work by Basch, Guibas and Zhang [5] has been done on proximity problems for moving points utilizing advanced data structures to optimize closest pair computation as well as maintaining a spanning tree with distance costs between the nodes. Their algorithm is easily modifiable to include distance metrics other than the Euclidean distance and is also quadratic in its runtime. Although they are not primarily concerned with the radio connectivity graphs that we are considering their algorithms and data structures can reasonably be modified to produce a radio connectivity graph.

This paper explores techniques for calculating graph connectivity between mobile nodes, which can incur lower computation costs, at a possible loss in accuracy. We investigate three simple techniques that can be used for such approximation. The quality of these approximations is evaluated using different metrics. One could evaluate the approximations by comparing the graphs that they give with respect to

the original connectivity graph. For instance, Chung et al. [7] propose a way to compare two graphs by breaking them into a minimal number of isomorphic partitions. We could compare the graphs given by the original connectivity and those with the approximations and compare the number of isomorphic partitions. However, it has been shown by Yao [18] that even checking if two graphs are 2-isomorphic is NP-complete. Instead, in this paper we rely on more modest structural measures of similarity for comparing our graphs, using certain metrics studied by previous researchers [2, 3] in addition to newer ones being proposed here. The main goal is to show that mobile network simulations using these graphs are going to produce close enough results to actual connectivity based simulations. Since the number of hops (for a message to get to a destination) and the interference (from other connections) are key determinants to message latencies, our graph comparison metrics attempt to capture these characteristics.

In addition to picking static graphs (i.e. the nodes are stationary), and comparing their connectivity similarity to the approximations, we also consider time-varying graphs (i.e. nodes are mobile using an appropriate mobility model), and examine how the differences manifest over time. We demonstrate that the connectivity approximation techniques can quite closely track the actual graph connectivity. We also show that one does not have to calculate connectivity at successive time steps in a network simulator, and we need to do this only occasionally at a coarser granularity. Even at these coarser intervals, we can use one of the approximation techniques, without having to ever calculate exact network connectivity. Such a technique has the potential to provide considerable speedup without introducing significant errors.

The rest of this paper is organized as follows. The next section goes over the different connectivity approximation methods after reviewing how the exact connectivity is itself calculated. Section 3 discusses different metrics that we use to compare the graph connectivities in both static and dynamic (time-varying) settings. In section 4, we tune some of the parameters for the methods, before giving experimental results in static and dynamic settings in section 5. Section 6 explores the possibility of not performing repetitive connectivity calculations within a network simulator. Finally, section 7 summarizes the contributions of this work.

2 Methods for Approximating Connectivity

As mentioned in the previous sections, our goal of connectivity calculations between clients in a mobile network is to obtain a graph where two clients are connected (an edge exists in the graph between these two vertices) if they are within radio range of each other. In the following discussion, we first discuss how connectivity would be computed if we are to get very close to the real world transmission/reception properties to not loose much accuracy (called real connectivity henceforth), and then study three different connectivity approximation methods. We assume that the radio range, R_a, that a client can transmit over is given as a parameter.

2.1 Real Connectivity

This model, which we call the real connectivity of the graph, is the most realistic of the connectivity methods we consider. To determine whether a pair of nodes is connected we simply look at the Euclidean distance between them, i.e. given two points $(x_i, y_i), (x_j, y_j)$, it is simply $sqrt(x_i - x_j)^2 + (y_i - y_j)^2$. If this distance is less than the radio range, then they are connected, otherwise they are not.

This model mirrors the two-ray ground propagation model, where received power is proportional to the inverse of the square of the distance between nodes up to a limiting distance and proportional to the inverse of the fourth power of the distance beyond that threshold. In the absence of interference this yields a connectivity range for each node that depends on the Euclidean distance between nodes. Because this model is not a perfect representation of reality and the radio interference is not accounted for our *real model* can be seen as an approximation itself. However, for the remainder of the paper, we will use this model as the base model for comparison.

Computationally (in terms of constants in the order notation) this model requires two subtractions, two multiplications, an addition and a comparison for each possible pair of nodes, of which there are $\binom{n}{2}$ given n nodes. This is assuming all links are symmetric, which occurs if the radio ranges for each node are equal. Note that we do not really need to take the square root since we can compare the square of the distance calculated with the square of the radio range, that can be calculated just once at the start of the simulation.

All of these calculations are made at every time-step, although there are methods of computing the changes to the node distance caused by taking into account the speed/granularity of node movements which can lower costs significantly. These calculations are also reasonably suited to parallelism if one is building the simulator (or at least performing these calculations) in parallel. One can simply dissect the space into quadrants and assign the resulting nodes in a quadrant to a processor to provide reasonable load balance (if the nodes are distributed uniformly over the space), though one cannot guarantee any bounds on load balance. If the quadrants are assigned such that their diameter is approximately the radio range then a given work unit will only have to communicate with the eight work units corresponding to the eight adjacent quadrants.

2.2 $l = 1$ Distance Metric Connectivity.

In this method, we simply replace the Euclidean distance metric with the $l = 1$, or Manhattan, distance metric. In this distance metric the distance between two points, $(x_i, y_i), (x_j, y_j)$, is calculated as the sum of the component-wise distances, $|x_i - x_j| + |y_i - y_j|$ (note that the differences in each dimension are to the power 1, and hence the name $l = 1$ while Euclidean distance uses $l = 2$). While one could again use R_a to check whether this distance falls within the bounds, we note that one could opt to have a difference range bounds (say R_{l1}), since a larger distance would be needed to get to the same radio range if one is

constrained by Manhattan movement. The trade-offs for different values of R_{l1} are evaluated later in this paper.

Though the number of calculations may be similar to the real connectivity calculation explained above, the constants associated with the calculations are lower (requiring only two subtractions and an addition). This scheme is also equally amenable to parallelism as in the previous case. The reason for considering this method (even though this may not be particularly accelerating the computation of the connectivity) is to find out how the accuracy changes when we make some approximations in the calculations of distance while still adhering to some of the basic properties associated with a legitimate distance metric (though not Euclidean).

2.3 Box Connectivity.

The next method we consider attempts to exploit the property that spatial proximity is more important in connectivity calculations, i.e. establishing connectivity only within a given neighborhood (box) and not bothering to check whether nodes not within this neighborhood are within the radio range.

To generate the box connectivity we split the two dimensional space into square sub regions of a given size, S_{box}. Let x_i, y_i denote the x and y coordinates of node i. Then node i belongs to box (α, β) iff $\lfloor \frac{x_i}{S_{box}} \rfloor = \alpha$ and $\lfloor \frac{y_i}{S_{box}} \rfloor = \beta$. Once we have the box assignment for each node we then say that a node is connected to every other node in the same box and every node in an adjacent box. That is each node in box (α, β) is connected to every node in boxes $(\alpha \pm 1, \beta \pm 1)$. Note that some of these adjacent boxes may not actually exist for boxes on the border of the simulation.

The computation required for this method is only two divisions for each node at each time step. With appropriate data structures, it is fairly straightforward to find out what other nodes one is connected to at each step. As in the previous cases, work partitioning for parallelism is straightforward, and is efficient if nodes are uniformly distributed in space.

2.4 k-means Cluster Connectivity.

The effectiveness of the box connectivity method described above is highly subjective to how the space is partitioned. Another approach to exploit spatial proximity is to analyze the spatial locations of objects and perform the calculations for the objects closest to a node. For instance, one can attempt to cluster the data, to identify different groups (clusters) of nodes, and only verify whether nodes in nearby clusters are in radio range (without having to check clusters farther away). We call such an approach cluster connectivity, or k-means method, since the k-means clustering algorithm [14, 13] is used to group the nodes.

To calculate the clustered connectivity we first apply the k-means clustering algorithm to our set of nodes to obtain k groups of nodes. We then say that a node can communicate with any node within its

cluster. We also allow a node to communicate with any node in a nearby cluster. To determine if a cluster is nearby we calculate the Euclidean distance between the centroids of the two clusters. If that distance is less than some range, R_{clus}, then the clusters are close enough, and thus all nodes in each cluster can communicate with each other. Again, we may need to tune R_{clus}, which can depend not only on the radio range, but also on the spatial characteristics of the nodes.

Initially calculating the cluster assignments may be quite computationally demanding, but once the initial clusters have been assigned finding new clusters at subsequent time steps can be much faster. This method allows a cluster based decomposition of the nodes in a parallel setting, providing a more load balanced mechanism for work partitioning even with nodes not being evenly distributed in space, unlike the earlier schemes.

2.5 A qualitative comparison

It should be noted that the choice of data structure employed has a considerable influence on the running time of each of the above methods, in addition to the distribution of the nodes within the spatial extent. In the worst case, an n^2 calculations may need to be performed, where even node is compared with every other node. However, one could maintain data structures such as keeping track of nodes that fall within different spatial regions (can be obtained by one pass through all the nodes taking $O(n)$ time), and having a node calculate connectivities only for the nodes within the regions which fall within its radio range (note that in the worst case, the data could get extremely skewed - all the data is one big cluster - making this perform very bad as well). Exploring the time costs for performing these computations, and data structures (e.g. [10, 15]) for implementing them efficiently, warrants a separate study by itself, that we intend to undertake in the future. In this paper, on the other hand, we focus mainly on the effect of the approximations on the graph (connectivity) structure as detailed in the rest of this discussion.

3 Graph Comparison Metrics and Measures

There has been a recent interest in studying the structure of real networks [2, 3]. We use similar techniques to compare the topology of radio connectivity graphs. The goal is to verify how close is the graph generated by the approximation methods compared to the one given by the real connectivity.

Towards this goal, we identify/examine different metrics about the graphs which in turn can have a bearing on the performance of the network in the simulation. In a broad sense, the important performance characteristics of a mobile network simulation are the number of hops (intermediate nodes) taken by a message to reach its destination (or even its ability to reach the destination in case the graph becomes disconnected) - a characteristic of the routing layer, and the congestion/interference experienced by the messages - a characteristic of the MAC layer. We propose a metric called the *percentage Hamming dis-*

tance that tries to compare the graphs in terms of its structure/paths that can give indications on the path length performance characteristics. We also examine the distribution of the shortest path between any pair of nodes to not only look at average behavior, but variances as well. Two of our metrics - degree distribution and clustering coefficient - are intended to capture the interference/congestion characteristics, since they give an idea of the density of the nodes in space.

It is not only important to pick a few graphs and show the effectiveness of the approximations on them. In an actual simulator, nodes will move, and we need to study how the behavior changes over time with such movement. So we also examine these metrics with a mobility model in place and we refer to these experiments as time-varying behavior.

3.1 Percentage Hamming Distance (%H)

Our first metric for graph similarity is based on the concept of Hamming distances. We can set up a connection matrix for each graph, where a 1 is placed in row i column j iff node i is connected to node j. We can then compute the Hamming distance between the two matrices (recall that the Hamming distance between two vectors is the number of positions in which they differ, for example $H([1001],[0101]) = 2$).

Hamming distance alone may not be a reliable metric and it is important to take the sizes of the graphs into consideration when comparing them. For example consider two pairs of graphs, the first pair has 2 nodes and the second pair has 5 nodes. In the first pair to be considered, one graph has the two nodes connected, the other does not, with the Hamming distance becoming 2 between their connectivity matrices (because the entries for node 1 being connected to node 2 appears as well as an entry for node 2's connection to node 1). In the second pair of graphs imagine that all of the connections are the same except for 2 connections, so we would find the Hamming distance to be 4. However, in the second pair the graphs are more similar than in the first pair, and the hamming distance may indicate otherwise. To fix this, we can divide the hamming distance by the total number of possible connections (if n is the number of nodes in the graph then this denominator will be n^2). We call this metric the *percent Hamming distance* and denote it as $\%H$. Smaller this value, the fewer the differences between the two graphs.

Now when we look at the previous example, we see that the $\%H$ of the first pair of graphs is $\frac{2}{4} = .5$ and the $\%H$ for the second set of graphs is $\frac{4}{25} = .16$. This fits much better with our intuition that the second pair is much more similar than the first. $\%H$ is targeted more at examining the routing characteristics of the approximation with that of the real connectivity.

3.2 Path Length Distribution

Even if the graphs may not resemble each other, what really matters is whether performance characteristics of the approximations match those of the real system. One important (and another routing) characteristic is the path length of a message to get to the destination. We could calculate the average

of all paths (from every source to every destination) in the network and compare the values in the two graphs. We could also calculate the distribution (number of paths having a certain length) which is a more detailed examination of the graphs, and this is what we use.

3.3 Degree Distribution.

The degree distribution of a graph has recently received widespread attention (see [6] for a very informative survey on this topic). We let the degree of a node be d and the number of nodes in a graph with degree d be defined as $|d|$. The degree distribution is simply a plot of d versus $|d|$. Work has been done to classify graphs using the degree distribution. Recently, a large amount of work has focused on scale free networks [2, 3], or graphs with a degree distribution that follows $|d| \sim d^{-\gamma}$. The random graph model of Erdős and Rényi [9] has been studied in great detail and is known to have a degree distribution that follows a Poisson distribution.

If the degree distribution of the approximation model matches that for the real connectivity graph, then it is an indication that the number of nodes that may potentially be contending/interfering with each other, as predicted by the model may be close to that of the actual connectivity (i.e. MAC layer aspects).

3.4 Clustering Coefficient.

Another widely studied graph measure is the clustering coefficient [17, 8, 4]. The clustering coefficient of node i is defined as the number of connections between its neighbors, n_i, divided by the total possible number of connections between them, which if node i has d_i neighbors is $\binom{d_i}{2}$. We consider the relationship between the degree of a node, d_i, and its clustering coefficient, $C_i = 2n_i/d_i(d_i - 1)$. This metric is similar to the degree distribution in the performance characteristics (MAC layer) that it is targeting.

3.5 Time Varying Behavior

Until now, we have not considered the graphs to change over time (i.e. the nodes are not moving). We consider the effect of approximations in connectivity calculations for moving nodes as well (referred to as time varying behavior). For all the time varying behavior experiments we used the random way point algorithm [12] for node mobility. We looked to capture two distinct types of time dependent behavior: how much inaccuracy is introduced in a graph by node mobility? and how do the characteristics of network performance change over time? Note that we can use the same metrics described above and see how these metrics change over time with respect to the real connectivity.

4 Parameter Tuning

Before we get to detailed experimental comparisons, we would like to first fix appropriate values for the parameters defined in each of the schemes. The best parameter values (to reduce errors) are not immediately obvious and we conduct experiments varying the values across an entire spectrum to get as close to the graph of the real connectivity. In these experiments, we use the graph difference metric, $\%H$, mentioned above. We then find minimum and maximum values for the parameters for each algorithm and ran experiments with the parameter values ranging uniformly between these extremes. We pick the value that gives the minimum $\%H$ from these experiments.

The parameters to be fixed include R_{l1} for $l = 1$, S_{box} for Box, and R_{clus} and k for k-means cluster connectivity. The minimum and maximum values for these parameters are chosen such that at one extreme there would be no false positives (i.e. we do not identify an edge from i to j in the approximate connectivity graph if there is no such edge in the real graph) and at the other there would be no misses (i.e. we do not miss out an edge that exists in the real graph). While these minimums and maximums can be calculated exactly for the $l = 1$ distance metric and box connectivity models, we use expected values for the k-means scheme (that could include a small number of false positives or misses at the two extremes).

Though we have conducted experiments with different parameters, the results are being explicitly shown for a 1 kilometer by 1 kilometer flat square region, with 1000 nodes and a real radio range, R_a, of 250 meters (which is used in subsequent comparisons as well).

The x-axis on the following graphs is the parameter we are tuning, and the y axis is the percent hamming distance between that graph and the exact one. In the case of the k-means clustering method there are two parameters to be tuned and thus the two horizontal axes are the parameters to be tuned and the vertical axis is again the percent hamming distance.

4.1 $l = 1$ Connectivity

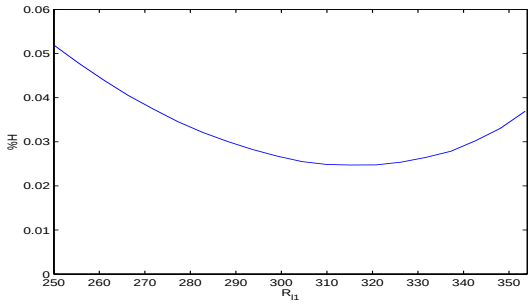


Figure 1: The parameter tuning results for the $l = 1$ distance metric connectivity method.

It can be easily verified that with a $R_{l1} \leq R_a$ no false positives will be present in the $l = 1$ distance metric method. Similarly if $R_{l1} \geq \frac{2}{\sqrt{2}} R_a$, no misses will occur. Using these bounds we see that we should let $R_a \leq R_{l1} \leq \frac{2}{\sqrt{2}} R_a$. The effect of varying R_{l1} in this range on the $\%H$ metric (computed between the graph given by $l = 1$ connectivity and that from the real connectivity) is given in Figure 1. We found that $R_{l1} \approx 1.234 * R_a$ gives the minimum $\%H$ (percent hamming distance). We use this value for the remaining experiments.

4.2 Box Connectivity

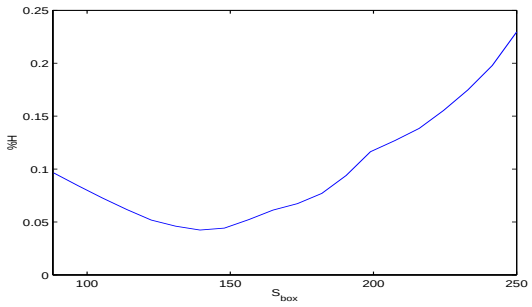


Figure 2: The parameter tuning results for the box connectivity method.

In this case, no false positives will be included if $S_{box} \leq \frac{R_a}{2\sqrt{2}}$. Similarly if $S_{box} \geq R_a$, no misses will occur. Consequently, we set $\frac{R_a}{2\sqrt{2}} \leq S_{box} \leq R_a$, and vary S_{box} uniformly in this range. The corresponding results are given in figure 2. We found that a value of $S_{box} \approx 0.5577 * R_a$ minimized the percent graph difference and is used for the remaining experiments.

4.3 k-means Clustering Connectivity

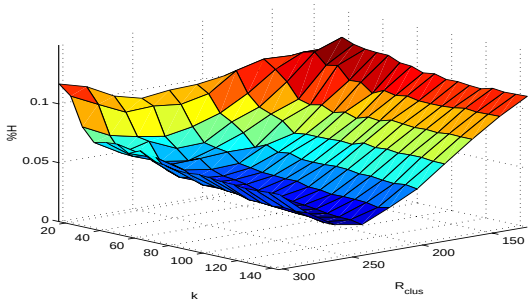


Figure 3: The parameter tuning results for the k-means connectivity method.

It is worth noting that the bounds for the two previous parameters are tight, i.e., if the minimum values for R_{l1} or S_{box} are set any higher then false positives may occur, and similarly if the maximum

values are any lower then misses may occur. Because of the nature of the k-means clustering algorithm such tight bounds are not easy for the clustered connectivity method. Instead, we set the limits for the clustered method based on average behavior. We assume that each cluster would have the average number of nodes per cluster and thus would cover on average a set percentage of the region.

It is also worth noting that as the number of clusters, k, approaches the number of nodes in the system this method breaks down into the actual connectivity model if $R_{clus} = R_a$. Because of this, instead of finding a value of k to minimize the percent graph difference we looked for a percolation point, i.e., we looked for an inflection in the graph where increasing the number of clusters starts producing diminishing returns.

We initially conducted experiments for R_{clus} and k in the range $R_a/2 \leq R_{clus} \leq R_a$ and $2 * (\frac{s}{R_a})^2 \leq k \leq N$, where s is the size of the region (1 Km in this case) and N is the number of nodes (1000 in this case). After inspecting the results we noted that the region we were concerned with was more in the region where $R_a/2 \leq R_{clus} \leq 1.25 * R_a$ and $(\frac{s}{R_a})^2 \leq k \leq 9 * (\frac{s}{R_a})^2$.

The results for parameter values varied uniformly in this region are given in figure 3. We find that the percolation point for the number of clusters occurs before $k = 4 * (\frac{s}{R_a})^2$ and the optimal value is for $R_{clus} = R_a$ (250 m in this case).

5 Experimental Results

All of the experimental results given here, use the same average node density, 1 node per 1000 square meters. The nodes are uniformly distributed over a square region. The real radio range, R_a, was chosen to be 250 meters. It is worth noting that increasing the node density has the same effect as running the experiments many times. As long as the density is set fairly high there is an extremely small amount of variation between runs. While several experiments were conducted to ensure consistency, results are presented for a single run.

We first present the $\%H$, path length distribution, degree distribution and cluster coefficient results with static graphs. We then present results for time varying behavior.

5.1 $\%H$ Results

Each of the approximate models is compared to the real model using percent hamming distance as defined earlier with the optimal parameters found in the parameter tuning experiments. The results for each model are shown in table 1 for a 4000 node system in a 2 Km by 2 Km space. We find overall the values to be quite small suggesting the graphs are somewhat similar. Of the three, we find that the $l = 1$ scheme comes the closest, since it does use a distance metric. Of the other two, the k-means scheme does a little better, though their differences are not very pronounced. These results suggest that these

	$l = 1$	Box	k-means
$\%H$	0.0075	0.0131	0.0128

Table 1: The $\%H$ results for each model compared to the real connectivity graph.

schemes may not be very bad in terms of the approximating the routing characteristics of the network compared to the real system. This will be reiterated in the next set of results examining the path lengths.

5.2 Path Length Distribution

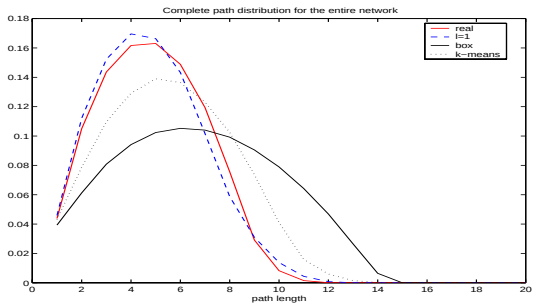


Figure 4: The of path lengths in the graph for each model.

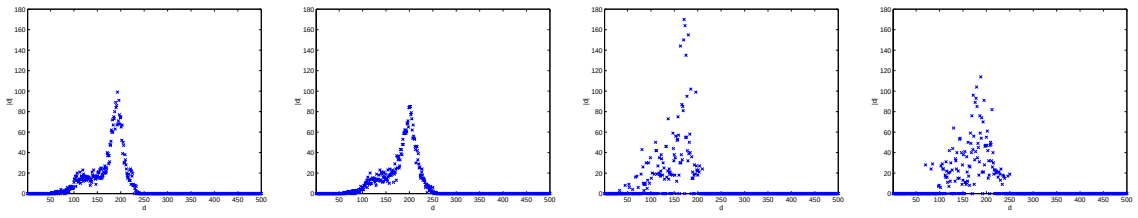
As mentioned earlier, the path length comparisons between the graphs provided by the models and the real connectivity can give indications on the accuracy of the routing characteristics. While one could look at an average of the path lengths between every pair of nodes, we give a more detailed depitiction by giving a distribution of the path lengths (the number of paths that are of a given length in the x-axis) in Figure 4. The results are again for a 4000 node system in a 2 Km by 2 Km square area.

These results again reiterate the observations we made in the $\%H$ results, in that the $l = 1$ metric closely tracks the routing characteristics of the real network. Of the other two, the box method seems to be much worse than the k-means method which may not be a bad approximation.

5.3 Degree Distribution

Having looked at the two metrics that are more indicative of routing characteristics, we next examine the congestion/interference characteristics by looking at the degree distribution (and clustering coefficient subsequently). These results are also for a 2 kilometer by 2 kilometer spatial extent with 4000 nodes to maintain the previously stated density.

The degree distribution results are given in Figure 5 for the three approximation together with those for the real connectivity. Note that the y-axis indicates the frequency of occurrence of a node with a degree specified on the x-axis. It is clear that the degree distribution is not similar to that of a scale free



(a) actual (b) $l = 1$ (c) box (d) k-means

Figure 5: The degree distributions for each connectivity method.

network. We might expect that these degree distributions would follow the simple Poisson distribution that is exhibited by random graphs [9]. Looking at the plot for the real connectivity model we can see that while there may be a Poisson component (around a mean of 200), the distribution may not be as clearly defined.

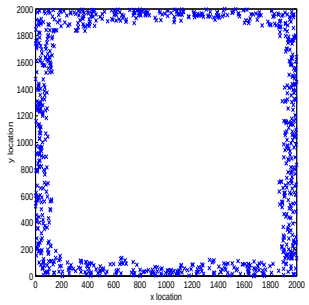
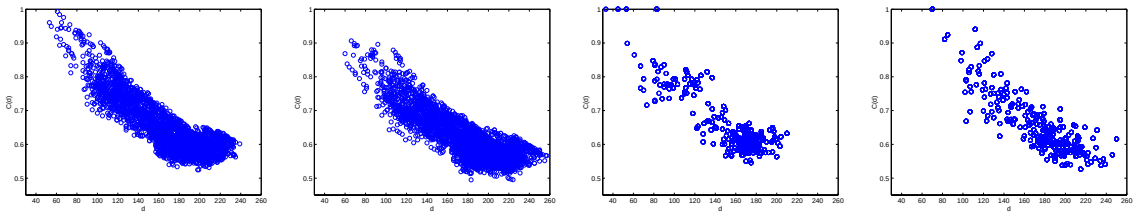


Figure 6: The locations of the set of nodes with node degree less than 150 in the real connectivity model .

For a closer investigation to find out why the distribution differs from the expected Poisson behavior, we tried to separately examine the nodes whose degrees were 150 or less (since the ones on the left side of this point are the ones causing the deviation from the anticipated behavior). We plot the spatial location of these nodes in the 2-dimensional space in Figure 6, and it is immediately clear why they are causing the discrepancy. These are the nodes near the edges of the considered region region, and as is to be expected they have fewer neighbors. It is to be noted that this discrepancy is a synthetic artifact made when we are bounding the spatial extent under consideration. When we are looking to scale large systems, covering a large spatial extent, these perturbations would decrease and the Poisson behavior would become more pronounced.

Comparing the real connectivity degree distribution to the other schemes, it is quite clear that the $l = 1$ model is quite close, as was the case for the routing characteristics earlier. In fact, the real and $l = 1$ graphs are the ones that really have a smoother curve, compared to the rest, since they are actually



(a) actual (b) $l = 1$ (c) box (d) k-means

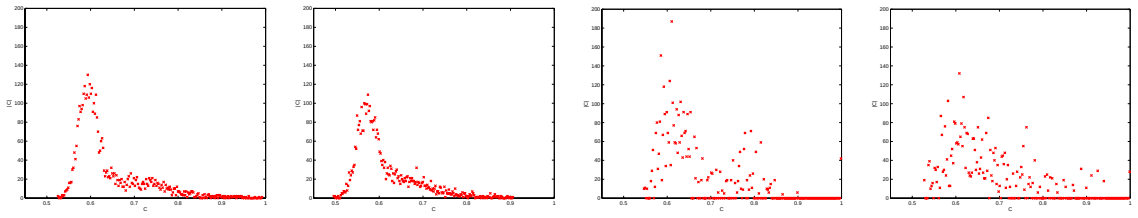
Figure 7: The Clustering coefficients for each connectivity method plotted against the node degree for each node.

computing a distance between any two points.

However, we would like to note that the box and k-means approaches (the latter in particular) are giving reasonable degree distributions. We can definitely see that the most frequently occurring degrees (and the overall mean) seems more or less in agreement with the real connectivity. Further, when we discount the nodes which are at the periphery of the spatial extent, the k-means approach is much more similar to the real connectivity.

5.4 Clustering Coefficient

The next metric we examine which again looks at the congestion/interference issue, is the clustering coefficient. The experiments were run on the same 2 kilometer by 2 kilometer square region as the degree distribution experiments, with 4000 nodes.



(a) actual (b) $l = 1$ (c) box (d) k-means

Figure 8: The distribution of Clustering coefficients for each connectivity method.

The results are given in figure 7 where the clustering coefficient ($C(d)$) on the y-axis is plotted for each d (the number of neighbors of a node) on the x-axis. The results show, as the degree distribution results did, that these graphs do not appear to exhibit the same structure that we would expect from either

a scale-free or hierarchical graph (where it has been observed to obey $C(d) \sim d^{-1}$ [8, 4]). We have also plotted the distribution of clustering coefficients in the network using each connectivity model in figure 8. As before, the $l = 1$ model gives the closest results to the real connectivity, though the behavior of box and k-means are not significantly different.

5.5 Time Varying Behavior

Having studied static graphs, we next move to evaluating the schemes with dynamic graphs, i.e. the nodes in the network move over time, and we examine the accuracy of the approaches over time. Instead of looking at all the earlier metrics, we specifically focus on one metric for routing (average path length) and one for congestion (mean node degree) characteristics. The random way point model [12] is used for moving the nodes.

The average path length over time results are given in figure 9. We first note that the average path length decreases with time, and this is an artifact of the random way point model. When a node comes to rest, it again picks a random destination, and increasing the likelihood of crossing the center region of the spatial extent. Over time, this effect, causes more nodes to be clustered towards the middle. A node in the middle has shorter path lengths on the average compared to nodes on the periphery, leading us to the observed behavior.

As before we find $l = 1$ closely tracking the path lengths in the real connectivity. However, k-means is not too far behind, and comes within 10% (or even lower) of the average path lengths of the real connectivity (and is much better than box connectivity).

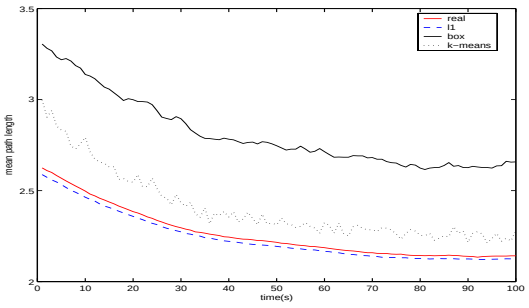


Figure 9: The average path length of the graph over time.

The mean node degree for the system over time with the random way point mobility is plotted in figure 10 for the different schemes. The same observation as in the previous case - more nodes tend to be in the center at any instant with time - causes the degree distribution to increase over time. In this case, we find that except for a few more jitters, the line for k-means follows that for the real connectivity much more closely than even $l = 1$, leading us to believe that this scheme can be effective in capturing both routing and MAC layer characteristics despite its approximations.

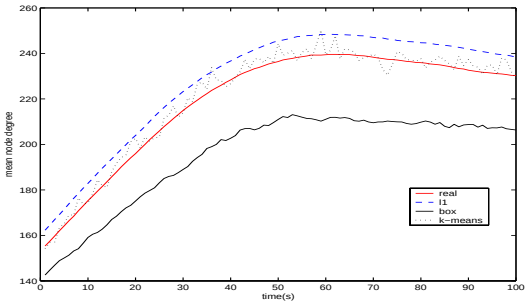


Figure 10: The average node degree of the graph over time.

6 Can we avoid repetitive connectivity calculations?

Please note that there are three kinds of costs involved in calculating connectivities in an actual simulator. The first is the calculation involved in finding out whether two nodes are connected (which is what $l = 1$ was trying to optimize compared to real). The second is the number of such calculations that may need to be performed (e.g. box or k-means can use spatial proximity to limit how many nodes you may need to check). Finally, in a mobile simulation, all these calculations need to be repeated at each time step (even in a discrete event simulator, time steps to account for every possible system event can be quite fine grain, causing a lot of overhead if these calculations have to be performed each time).

Having looked at schemes for the first two costs, we now examine the third aspect - do we really need to recalculate connectivities between nodes during each fine time step in the simulation? Rather, we would like to generate the graph every "once in a while", and live with simply use this graph for successive time steps. We call the time interval that the simulator runs on as the "fine grain" time interval and call the graph update interval as the "coarse grain" interval.

Since we would like to not have to update the graph during every "fine grain" interval we would like to study the consequences of such an action. To do this we chose a fine grain interval on which to run the real connectivity model, i.e. the graph connectivity is updated at each fine grain interval - let us refer to this as *actual connectivity* in this section. We then ran all of the mobility models (even the real one) on a coarse grain interval, and did not re-calculate their connectivities until the next coarse grain interval. We then compare the graphs of all these schemes (including the real which actually becomes an approximation in this case because it is only done at coarse grain intervals) with the graph of the actual connectivity as defined above. We can examine the resulting effect on any of the described metrics, though we specifically present results for the $\%H$ metric.

The experiments were run on a 1 kilometer by 1 kilometer are with a 1000 nodes. The random way point mobility model was used with a maximum speed of 5 meters/second and a pause time of zero seconds (i.e. nodes started towards their next destination as soon as they reached their current one). The fine grain time interval was 1 second and each experiment was run for 10 coarse grain time intervals. We

present results for coarse grain intervals of 50 seconds, and a longer 250 seconds. Note that the velocities chosen are fast enough for a reasonable number of changes to occur in connectives even in the 50 second coarse grain interval.

In general we would expect to see a saw tooth pattern in a graph of $\%H$ vs. time. Specifically, when the connectivity models are updated at the coarse grain interval boundaries, they should be very close to the actual connectivity (the real one should match the actual, giving a $\%H$ of 0). As the network changes over the fine time granularities, we would expect that they would become less and less accurate ($\%H$ increases) until the beginning of the next coarse grain time interval.

We observe these effects in Figure 11 where the $\%H$ is plotted as a function of time. We notice that despite the rise in accuracy (which is present even in real connectivity), even at 50 seconds after the last connectivity computation, the differences from the actual network may not be very significant. Note that (by observing this graph closely) we could bound $\%H$ values to less than 0.06 with 20 second coarse grain calculations - which implies a *20 fold speedup in connectivity calculations* compared to doing this at every fine step.

As can be expected, the errors grow with larger coarser time intervals which is evident in the results for a 250 second interval in Figure 12. In these cases, even real turns out much worse, and in fact with long enough intervals it is possible that one of our approximations can turn out to be more accurate than real (we observed this though this may not be explicitly decipherable in the graphs).

Once again we find that the approximation mechanisms can be used even for the coarse gain intervals of say 20 seconds (to give a speedup of 20), without every having to compute real connectivity, and keeping the errors quite small.

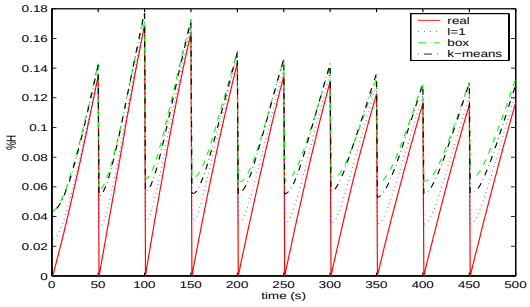


Figure 11: The mobility results using a short (50 seconds) coarse grain time interval.

7 Concluding Remarks

This paper has investigated an important component in the simulation of large scale mobile networks that involves the calculation of the connectivity between the nodes. Since this operation is needed at

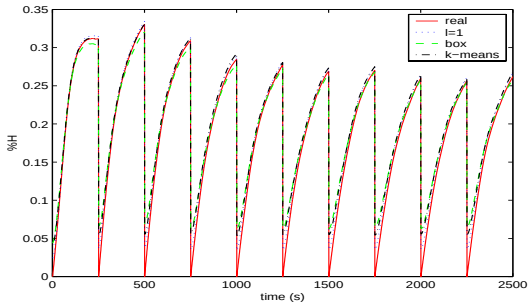


Figure 12: The mobility results using a long (250 seconds) coarse grain time interval.

every time step of the simulation, and its cost can go up quadratically with the number of nodes, it is very important to examine different ways of optimizing the calculations - simplifying the math involved in the calculations, reducing the number of pairs between which the calculations need to be performed, and avoiding the calculations themselves as far as possible - while not compromising on accuracy.

We have specifically examined three specific methods for approximating the connectivity between the nodes, while preserving the macroscopic properties of the network under consideration that affect its routing layer (the number of hops) and MAC layer (the congestion/interference) performance. In addition, this paper has also proposed different metrics by which the graph connectivities given by these approximations can be evaluated in terms of the accuracy of routing and MAC layer characteristics.

We find that despite the differences in how distance is calculated with the $l = 1$ method, the very fact that some notion of distance is used makes this method perform quite well across the different metrics considered. While this method may save some basic operations in the calculations, its complexity is not significantly different from that for calculating the real connectivity. Hence, its applicability in practice may be limited. Of the other two methods, we find that the k-means approach is better than the box connectivity overall. In terms of the routing characteristics, k-means does a fairly good job in terms of the path lengths. It may not be a bad choice even for MAC layer characteristics since we found that the average node degree turned out to be even better than $l = 1$. We find that these observations hold not just for static graphs, but also when we track the trends over time with a mobility model in place.

Another contribution of this paper is in showing that one does not have to repeatedly calculate connectivities at each time step in the simulation. We show that even these approximation can be calculated once in a while (as low as once every 20 time steps), without ever having to calculate the real network connectivity, and still keep errors relatively small.

It is to be noted that this paper has taken a preliminary step in the goal of optimizing and evaluating connectivity calculations in mobile network simulations. In addition to developing more approximation methods and designing a comprehensive suite of graph accuracy evaluation metrics (that are important to a network simulator), there is an important aspect of this work that we need to investigate in the future.

This is the evaluation of these schemes in an actual network simulator to evaluate their benefits (in terms of time) and their loss in accuracy (on the actual network statistics being studied). However, there is no current infrastructure readily available for such an undertaking. Most current network simulators do not allow for a connectivity graph to be given as input to the simulator [19, 11]. Rather, they require spatial coordinates of each node to be specified, and the simulator itself calculates the connectivity. It is a non-trivial problem to generate spatial coordinates of nodes from a connectivity graph. Enabling simulation technologies that directly take connectivity graphs is itself part of our future research agenda. The simulator can then further refine this model taking into account physical transmission characteristics, terrain information, etc.

Before we conclude, we would like to point out that loss in accuracy is inevitable if we do not want to calculate the connectivity between nodes in a network simulator exactly. However, it is our goal to strive for a reasonable approximation of the interactions between the mobile nodes so that they can be computed efficiently while still capturing the dynamics of the system on a macroscopic level, since we believe that this may be the only option when we are looking to simulate very large networks (of the order of a million nodes or higher).

References

[1] R. Albert, A.-L. Barabási. *Statistical Mechanics of Complex Networks.* Reviews of Modern Physics, 2001. (`citeseer.nj.nec.com/442178.html`)

[2] A.-L. BARABÁSI AND R. ALBERT, 1999, *Emergence of scaling in random networks. Science* 286, 509-512.

[3] A.-L. BARABÁSI, R. ALBERT AND H. JEONG, 1999, *Mean-field theory for scale-free random networks. Physica A* 272, 173.

[4] A.-L. BARABÁSI AND E. RAVASZ, *Heirarchical organization in complex networks*, preprint May 30 2002.

[5] J. BASCH, L. J. GUIBAS, AND L. ZHANG, *Proximity Problems on Moving Points, in Proceedings of 13th ACM Symposium of Computational Geometry*, 1997.

[6] B. BOLLOBÁS, 1985, *Random graphs,* Academic Press, London-New York, 1985.

[7] F. R. K. CHUNG, P. ERDÖS, R. L. GRAHAM, S. M. ULAM, AND F. F. YAO, 1979, *Minimal decomposition of two graphs into pairwise isomorphic subgraphs, in Proceedings of the Tenth Southeastern Conference on Combinatorics, Graph Theory, and Computing*, Boca Raton, Florida, April, 1979.

[8] S. N. DOROGOVTSEV, A. V. GOLTSEV AND J. F. F. MENDES, 2001, *Los Alamos Archive cond-mat/0112143.*

[9] P. ERDŐS AND A. RÉNYI, 1959, On random graphs-i *Publ. Math. Debrecen* 6, 290.

[10] E. FORGEY, *Cluster analysis of multivariate data: Efficiency vs. interpretability of classification, Biometrics*, 21:768, 1965.

[11] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. GloMoSim: A Scalable Network Simulation Environment. UCLA Computer Science Department Technical Report 990027, May 1999.

[12] D.B. JOHNSON AND D.A. MALTZ, *Dynamic Source Routing in Ad Hoc Wireless Networks*, in Mobile Computing, edited by T. Imielinski and H. Korth, chapter 5, pp.153-181, Kluwer Academic Publishers, 1996

[13] T. KANUNGO, D.M. MOUNT, N.S. NETANYAHU, C. PIATKO, R. SILVERMAN, A. Y. WU, *The Analysis of a Simple k-Means Clustering Algorithm Symposium on Computational Geometry*, 2000

[14] S.P. LLOYD, *Least squares quantization in PCM, IEEE Trans. Inform. Theory*, vol IT-28, pp. 129 137, Mar. 1982.

[15] J. MATOUSEK, *On Approximate Geometric k-Clustering, Manuscript, Department of Applied Mathematics, Charles University*, Prague, Czech Republic, 1999.

[16] C. SANTIVÁÑEZ, B. MCDONNALD, I. STAVRAKAKIS, AND R. RAMANATHAN *On the Scalability of Ad Hoc Routing Protocols, in Proceedings of IEEE INFOCOM'02*, New York, June, 2002.

[17] D. J. WATTS AND S. H. STROGATZ, 1998, *Collective dynamics of 'small-world' networks. Nature* 393, 440.

[18] F. FRANCIS YAO, 1979, *Graph 2-isomorphism is NP-complete, Information Processing Letters 9: 68-72.*

[19] *ns-2* Network Simulator. www.isi.edu/nsnam/ns, 2000 (version 2.1b8a)